ESD-TR-67-328

Technical Report

435

D. Esterling

LCAO Secular Determinant Program

12 July 1967

Prepared under Electronic Systems Division Contract AF 19(628)-5167 by

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Lexington, Massachusetts

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology, with the support of the U.S. Air Force under Contract AF 19(628)-5167.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

LCAO SECULAR DETERMINANT PROGRAM

D. ESTERLING

Group 83

TECHNICAL REPORT 435

12 JULY 1967

LEXINGTON MASSACHUSETTS

ABSTRACT

This report describes a computer program designed to set up the secular determinant arising from an energy band calculation in the LCAO (Linear Combination of Atomic Orbitals) approximation. The program determines which transfer integrals vanish, and which are related; further, it computes the appropriate structure factor (which contains the momentum dependence) for each entry in the secular determinant. This program can handle arbitrary crystal symmetry, unit cells with many inequivalent atoms, interactions involving up to fourth-nearest neighbors, and a choice of s-, p-, and/or d-orbitals on the various inequivalent atoms. The transfer integrals are left as parameters to be determined from the eigenvalues corresponding to special symmetry points in the Brillouin zone.

Accepted for the Air Force
Franklin C. Hudson
Chief, Lincoln Laboratory Office

CONTENTS

LCAO SECULAR DETERMINANT PROGRAM

This report describes a computer program designed to set up the secular determinant arising from an energy band calculation in the LCAO (Linear Combination of Atomic Orbitals) approximation. The program can handle arbitrary crystal symmetry, unit cells with many inequivalent atoms, interactions involving up to fourth-nearest neighbors, and a choice of s-, p-, and/or d-orbitals on the various inequivalent atoms. The transfer integrals are left as parameters to be determined from the eigenvalues corresponding to special symmetry points in the Brillouin zone.* These eigenvalues may be obtained, for example, by an augmented plane wave (APW) calculation.

The input data must specify the allowed symmetry operations of the crystal, the three non-coplanar lattice translations, the positions of the inequivalent atoms, the orbitals to be associated with these atoms, and the relevant interactions. The input is discussed in Sec. II; the output is discussed in Sec. V.

In principle, the secular determinant could be written down immediately if all the transfer integrals were distinct. However, certain transfer integrals vanish, others are identical (to within a sign), and still others may be specified as linear combinations of remaining ones. The various symmetry operations of the crystal generally carry a given transfer integral into a linear combination of other transfer integrals. This situation arises when an orbital transforms into a linear combination of orbitals. These operations yield relationships or equations involving the transfer integrals, and may be used to reduce the number of paramcters to a minimal set. A particularly simple example occurs when a transfer integral is transformed into the negative of itself. It then vanishes.

At this point, it will be convenient to list certain variable names which recur throughout the program.

NA	Specifies the number of inequivalent atoms in unit cell.
NS	Specifies the number of symmorphic symmetry operations (. SSO.).
NNS	Specifies the number of nonsymmorphic symmetry operations (. NSSO.).

* This technique was first suggested by J.C. Slater and G.F. Koster, Phys. Rev. 94, 1498 (1954).

TRS(I, J) A 3 × 3 matrix with each row specifying a non-eoplanar latticc translation* (the numbcr in the first column specifics the x-coordinate, etc.). I indicates the row, and J indicates the eolumn.

SYM(I, J, K) A matrix spccifying a given symmetry operation. 1 indieates the symmetry operation under study. For a fixed value of 1 (that is, for a given symmetry operation), J and K indieate the row and column of a 4 × 4 submatrix that cxpresses explicitly the given symmetry operation. This matter is explained in morc detail in See. III.

N(I, J) Speeifics thc extcnt of interaetions eonsidercd bctween I and J, where I and J indicate two incquivalent atoms; e.g.,

If N equals 0, then self-interactions only are considered (if relevant)

If N equals 1, then interactions between atom I and all thc nearest atoms of type J are to be considcred

If N equals 2, then atom I to ncxt-nearest atoms of type J are to be eonsidered

Etc.

NSYM(I) Explained along with subroutinc NONSYM – if no atoms arc related by an . NSSO., then all NSYM(I) = 0.

NORB(I) Specifies type of orbitals to be studicd; e.g.,

If NORB(I) = 1, then only s-type orbitals on atom I are considered

If NORB(I) = 2, then s- and p-typc orbitals on atom I are eonsidered

If NORB(I) = 3, then s-, p-, and d-type orbitals on atom I are considered.

NCHEM(I) Specifies ehemical nature of atom I; e.g.,

NCHEM = 1, atom of chemical type 1

NCHEM = 2, atom of chcmical type 2

NCHEM = 3, atom of chcmieal typc 3.

NA1 Specifies number of inequivalent atoms of ehemical type 1.

NA2 Specifies number of inequivalent atoms of chcmical type 2.

NA3 Specifies number of inequivalent atoms of chemieal type 3.

(NA1 + NA2 + NA3 = NA) .

* Any distances supplied to the program shauld be in some appropriate reduced units, e.g., in multiples af a, where a is any edge length af the unit cell.

NSM(I, J) — A matrix specifying the allowed . SSO., where I indicates atom number I, and J indicates . SSO. number J.

NSM(I, J) = 1 implies that J^{th} . SSO., with origin at atom I, is allowed*

NSM(I, J) = 0 implies that J^{th} . SSO., with origin at atom I, is not allowed.

CLST — Specifies closest distance between any two atoms in lattice. It is used to compensate for rounding off errors in machine. One only needs to specify a rough estimate of CLST. Internally, CLST is reduced by a factor of 100.

NL — Specifies atom number NL. Since the program deals with pairs of atoms when considering transfer integrals, generally one atom is chosen (in turn) and then its interactions with the other atoms are considered. Conventionally, the former atom is associated with the label NL and is referred to as the "fixed" atom in this report, while the latter atom is associated with the label NM and is referred to as the "variable" atom. Consult Sec. II, part (b) for more details on this point. If there are no . NSSO., then the numbering system chosen may be arbitrary; however, if there is an . NSSO., the numbering system is fixed by this choice. This point is amplified in Sec. IX.

LL — Specifies type of orbital on NL [see NORB(I)]

LL = 1, s-type orbital
LL = 2, p-type orbital
LL = 3, d-type orbital.

LLM — Specifics which orbital, given the type, is being considered on NL.

For LL = 1 and LLM = 1 , s-orbital is under study
For LL = 2 and LLM = 1 , p_x-orbital is under study
LLM = 2 , p_y-orbital is under study
LLM = 3 , p_z-orbital is under study
For LL = 3 and LLM = 1 , d_{xy}-orbital is under study
LLM = 2 , d_{xz}-orbital is under study
LLM = 3 , d_{yz}-orbital is under study
LLM = 4 , $d_{x^2-y^2}$-orbital is under study
LLM = 5 , d_{z^2}-orbital is under study.

NM — Specifies atom number NM.

ML — Specifies type of orbital on NM.

* Although a lattice possesses a certain group symmetry, all the point operations of that group will carry the lattice into itself only at certain positions. At other points, the symmetry will be more restricted. At a general point, there will be no symmetry.

MLM	Specifies which orbital, given the type, is being considered on NM.
NX	Specifies which level of atoms of inequivalent type NM. There are whole classes of atoms associated with the number NM: those which are closest to our "fixed" atom NL (corresponding to an NX value of 1), and those which are the next closest to atom NL (corresponding to an NX value of 2), etc.
NLEV	Specifies which atom of type NM within a given level (or fixed value of NX) is being studied.

There are limitations on certain of the above variables:

$$\begin{aligned} NA &\leqslant 18 \\ NA1 &\leqslant 6 \\ NA2 &\leqslant 6 \\ NA3 &\leqslant 6 \\ NS &\leqslant 48 \\ NNS &\leqslant 1 \\ N &\leqslant 4 \\ NCHEM &\leqslant 3 \\ NLEV &\leqslant 10. \end{aligned}$$

These limitations may be modified by changing the appropriate initialization statements involved,*† so that the program conforms to the particular problem under consideration and to the storage capacity of the system used.

In the following sections, we shall consider each of the subroutines in more detail.

I. LCAO

The calling sequence of the program is indicated in Fig. 1, where the subroutine names are displayed. LCAO reads in the input data using READ, and calculates the effect of the various point symmetry operations (rotations and reflections) on the s-, p-, and d-orbitals in ORBIT. LCAO then calls POISON, which computes the positions of all atoms considered in the particular program.

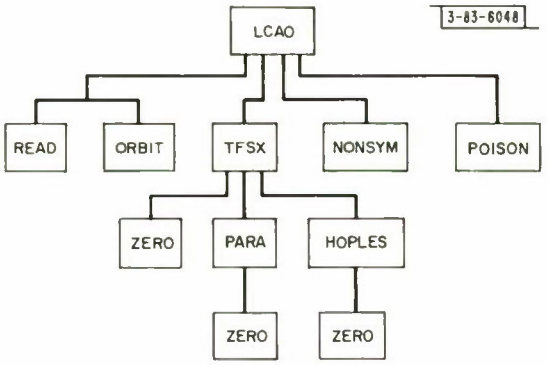

Fig. 1. Calling sequence of subroutines.

* These initiolizations are for ITF in LCAO, and for KKW and MISW in ZERO.

† NNS and NCHEM would require more elaborate modifications.

Following this, LCAO sets up the correspondence between the transfer integrals and their Hermitian conjugate (H.C.). A transfer integral is specified by the set of values: NL, LL, LLM, NM, ML, MLM, NX, and NLEV. The H.C. transfer integral would be specified by a set of values: NM, ML, MLM, NL, LL, LLM, NXHC, and NLVHCJ – where NXHC and NLVHCJ are to be determined (see Fig. 2). As indicated in Fig. 2, the position of the atom of type NL (associated with the labels NXHC, NLEVHC) relative to the atom of type NM is known; it is the negative of the position of the atom of type NM (associated with the labels NX, NLEV) relative to the atom of type NL. Using this information, LCAO locates the atom of type NL in the corresponding position. This then determines the values of NXHC and NLVHCJ.

This information is to be used by the program to reduce the number of independent parameters, using the equality between a transfer integral and its H.C. Here, one should recall, the program keeps one atom "fixed" when relating and parameterizing transfer integrals. In so doing, it generates relations between transfer integrals only in a particular row in the secular determinant. However, a transfer integral may be related to some other transfer integral in the same column but different rows (i.e., NM considered "fixed").

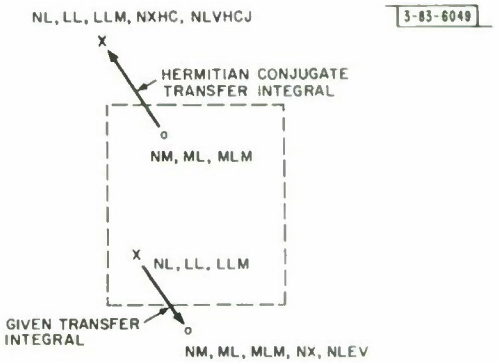

Fig. 2. Relation between given transfer integral and its Hermitian conjugate. Dashed line corresponds to unit cell boundary.

Now, the given transfer integral is equal to its H.C. Further, the H.C. lies in the NM row, which contains the same relationships among its transfer integrals that the NM column contains among its corresponding transfer integrals. Hence, we can use the above equality to search for relationships in columns as well as in rows.

Following the above calculations, LCAO then begins setting up the secular determinant. LCAO chooses a particular entry in the secular determinant in the following expeditious order: LL, ML, chemical type of "fixed" atom, chemical type of "variable" atoms, NL, NM, LLM, and MLM. A given transfer integral, within a given entry, is specified by NX and NLEV. This specification is done either in TFSX or in NONSYM.

Finally, LCAO calls either TFSX or NONSYM. Ordinarily, TFSX will be called; however, if the given entry is related to some previously considered entry by an .NSSO., then NONSYM is called. The variable NSM determines which subroutine is called. Its use is further explained in the description of NONSYM (Sec. IX).

II. READ

Following is the order in which the data cards are read in using subroutine READ.

(a) One card with the values of NA, NS, and NNS (integer format*). These values are to be right adjusted in columns 1 to 10, 11 to 20, and 21 to 30; i.e., if NA = 1, then put a "1" in column 10; if NA = 10, then put a "1" in column 9 and a "0" (or blank) in column 10.

* Integer format – use no decimal points, right adjusted; decimal format – use decimal point, put number anywhere in columns reserved.

(b) NA cards listing the positions of the NA inequivalent atoms (in the assigned order). Each card contains the x-, y-, and z-coordinates* of the atom in columns 1 to 10, 11 to 20, and 21 to 30. This is in decimal format.† Note that card number 1 should have three 0's since, by convention, the first atom fixes the origin for our coordinate system.

(c) Three cards specifying each of the three non-coplanar lattice translations. Each card contains the x-, y-, and z-components in the same decimal format as above.

(d) Four × NS cards specifying the NS . SSO.'s. Each set of four cards specifies one . SSO. — these are in the form of 4 × 4 matrices (see Fig. 3 in Sec. III for more details). The first row of the . SSO. matrix is on the first card in the set, the second is on the second card, etc. Again, the same decimal format, now (with four numbers) using columns 1 to 10, 11 to 20, 21 to 30, and 31 to 40.

(e) Four × NNS cards specifying the . NSSO. in the same way as in (d) above. If NNS = 0, then, of course, no cards are to be supplied.

(f) NA cards specifying the array N. There are NA entries on each card. If we think of N as an NA × NA matrix, then the first card contains the first row, etc. The cards are in integer format, but 3 columns for each entry instead of 10 (using columns 1 to 3, 4 to 6, 7 to 9, etc.).

(g) One card with NA entries specifying NSYM (see Sec. IX for more details). If there is no . NSSO., this card may be blank. The format is the same as (f).

(h) One card with NA entries specifying NORB; same format as (f).

(i) One card with NA entries specifying NCHEM; same format as (f).

(j) One card with three entries specifying NA1, NA2, and NA3; same format as (a).

(k) NA cards each with NS entries specifying the array NSM. Again, considering NSM as an NA × NS matrix, the first card corresponds to the first row, etc. The cards are in integer format, but 1 column for each entry.

(l) One card specifying CLST. The format is decimal; the columns reserved are 1 to 10.

III. ORBIT

As mentioned previously, ORBIT computes the effect of a given symmetry transformation on a given orbital.

The transformations are read in as 4×4 matrices of the form shown in Fig. 3. (R is a 3 × 3 matrix representing the associated point transformation, where $\sum_{j=1}^{3} R_{ij} x_j = x'_i$. The subscripts i and j label the Cartesian coordinates; a, b, and c are the three (Cartesian) components of the associated nonlattice translation. All transformations are active, i.e., one rotates the atoms, not the coordinate system.) If the transformation is an . SSO., then a = b = c = 0. The 3 × 3 matrix in the upper left-hand corner is then a conventional matrix representing the transformation in Cartesian coordinates, e.g., reflection in x-y plane would have two entries of unity

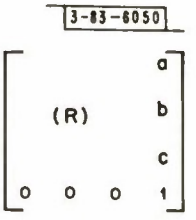

Fig. 3. Form of matrices representing given symmetry operation.

* Any distances supplied to the program should be in some appropriate reduced units, e.g., in multiples of o, where o is any edge length of the unit cell.

† Integer format — use no decimal points, right adjusted; decimal format — use decimal point, put number anywhere in columns reserved.

along the diagonal, followed by an entry of minus unity, with all other entries 0 (in the 3×3 matrix). Before the representation of an . NSSO. is described, the term . NSSO. should be defined, since it differs from the usual convention.

The term is best explained by an example. The diamond structure has 48 symmetry operations: 24 . SSO., and 24 . NSSO., where . NSSO. means nonsymmorphic symmetry operation in the conventional sense. Any one of the operations corresponds to a rotation and/or reflection followed by nonlattice translation. If we chose any one . NSSO. in the conventional sense and combined this operation with each of the 24 . SSO., then all 24 . NSSO. would be generated in the conventional sense. In this report, an . NSSO. corresponds to any one of the conventional . NSSO. Further, the a, b, c, in the 4×4 matrix representation of this . NSSO., correspond to the x-, y-, and z-components of the associated nonlattice translation, respectively, while R (the 3×3 matrix) is the associated point transformation.

Now, in general, a given orbital transforms into a linear combination of orbitals of the same type, either under an . SSO. or an . NSSO. (of course, the orbital is unaffected by the values of a, b, and c for an . NSSO.). Hence, a given transformation A generates a set of 1×1, 3×3, and 5×5 dimensional matrices indicating how the s-, p-, and d-orbitals transform, respectively. The form of each of these matrices will now be considered.

The orbitals are defined in terms of the usual polar coordinate as:

$$s = 1/2\sqrt{\pi}$$

$$p_x = (\sqrt{3}/2\sqrt{\pi}) \sin\theta \cos\varphi$$

$$p_y = (\sqrt{3}/2\sqrt{\pi}) \sin\theta \sin\varphi$$

$$p_z = (\sqrt{3}/2\sqrt{\pi}) \cos\theta$$

$$d_{xy} = (\sqrt{15}/2\sqrt{\pi}) \sin\theta \cos\varphi \sin\varphi$$

$$d_{xz} = (\sqrt{15}/2\sqrt{\pi}) \sin\theta \cos\theta \cos\varphi$$

$$d_{yz} = (\sqrt{15}/2\sqrt{\pi}) \sin\theta \cos\theta \sin\varphi$$

$$d_{x^2-y^2} = (\sqrt{15}/4\sqrt{\pi}) \sin^2\theta (\cos^2\varphi - \sin^2\varphi)$$

$$d_{z^2} = (\sqrt{15}/4\sqrt{\pi}) (3\cos^2\theta - 1)$$

$$= (\sqrt{15}/4\sqrt{\pi}) (2\cos^2\theta - \sin^2\theta \cos^2\varphi - \sin^2\theta \sin^2\varphi) \quad .$$

Under a transformation A, s-orbitals transform into orbitals of type s ($s \xrightarrow{A} s$). Therefore, the 1×1 matrix contains only the entry unity.

The transformation of the p-orbitals is obvious, e.g.,

$$p_x \xrightarrow{A} A_{11}p_x + A_{12}p_y + A_{13}p_z \quad \text{etc.}$$

Therefore, the 3×3 matrix is just the matrix A.

For the d-orbitals, the transformation is not as obvious,* e.g.,

$$c\,|xy\rangle \xrightarrow{A} c \sum_{i,j} A_{1i}A_{2j}\,|ij\rangle = c(A_{11}A_{22} + A_{12}A_{21})\,|xy\rangle$$
$$+ c(A_{11}A_{23} + A_{13}A_{21})\,|xz\rangle + c(A_{12}A_{23} + A_{13}A_{22})\,|yz\rangle$$
$$+ (c/2)\,2A_{11}A_{21}\,|x^2\rangle + (c/2)\,2A_{13}A_{23}\,|z^2\rangle$$
$$+ (c/2)\,2A_{12}A_{22}\,|y^2\rangle$$

where $c = \sqrt{15}/2\sqrt{\pi}$ is the normalization constant for the d_{xy}-orbital which we have explicitly separated out. We have proper normalization for the ket's corresponding to d_{xz} and d_{yz} automatically, since d_{xy}, d_{xz}, and d_{yz} all have the same normalization constant. However, the ket's corresponding to the $d_{x^2-y^2}$- and d_{z^2}-orbitals do not have proper normalization. (Recall that this normalization constant is $c/2 = 15/4\sqrt{\pi}$.) Now, under a given operation, we will get some linear combination $d\,|x^2\rangle + e\,|y^2\rangle + f\,|z^2\rangle$, which can be expressed as

$$a\,|x^2 - y^2\rangle + b\,|3z^2 - r^2\rangle \quad .$$

This is obviously not true for an arbitrary matrix, but is true for a representation of crystal rotations and/or reflections. Now, if we require

$$dx^2 + ey^2 + fz^2 = a(x^2 - y^2) + b\left[\frac{1}{\sqrt{3}}(2z^2 - x^2 - y^2)\right]$$

then

$$a - \frac{b}{\sqrt{3}} = d$$
$$-a - \frac{b}{\sqrt{3}} = e$$
$$\frac{2b}{\sqrt{3}} = f \quad .$$

Since we have three equations for two unknowns (a and b), we must have an equation relating d, e, and f so that these equations are consistent. The form of the three equations indicates that this consistency equation is $d + e = -f$. Hence, $a = (d - e)/2$ and $b = (\sqrt{3}/2)\,f$.

Returning to the equation indicating the transformation of d_{xy} under A, we see that $d = 2A_{11}A_{21}$, $c = 2A_{12}A_{22}$, and $f = 2A_{13}A_{23}$ (separating out the normalization constant c/2). Hence, $a = A_{11}A_{21} - A_{12}A_{22}$, and $b = \sqrt{3}\,A_{13}A_{23}$. We then have

$$d_{xy} \xrightarrow{A} (A_{11}A_{22} + A_{12}A_{21})\,d_{xy}$$
$$+ (A_{11}A_{23} + A_{13}A_{21})\,d_{xz}$$
$$+ (A_{12}A_{23} + A_{13}A_{22})\,d_{yz}$$

* An orbital denoted as the ket is unnormalized; that is, it is defined exactly as the corresponding d-orbital without the normalization constant.

$$+ (A_{11}A_{21} - A_{12}A_{22})\, d_{x^2-y^2}$$

$$+ \sqrt{3}\, A_{13}A_{23}\, d_{z^2}$$

$$d_{xz} \xrightarrow{A} (A_{11}A_{32} + A_{12}A_{31})\, d_{xy}$$

$$+ (A_{11}A_{33} + A_{13}A_{31})\, d_{xz}$$

$$+ (A_{12}A_{33} + A_{13}A_{32})\, d_{yz}$$

$$+ (A_{11}A_{31} - A_{12}A_{32})\, d_{x^2-y^2}$$

$$+ \sqrt{3}\, A_{13}A_{33}\, d_{z^2}$$

$$d_{yz} \xrightarrow{A} (A_{21}A_{32} + A_{22}A_{31})\, d_{xy}$$

$$+ (A_{21}A_{33} + A_{23}A_{31})\, d_{xz}$$

$$+ (A_{22}A_{33} + A_{23}A_{32})\, d_{yz}$$

$$+ (A_{21}A_{31} - A_{22}A_{32})\, d_{x^2-y^2}$$

$$+ \sqrt{3}\, A_{23}A_{33}\, d_{z^2} \quad .$$

In a similar fashion, we get from

$$k\, |x^2 - y^2\rangle \xrightarrow{A} k \sum_{i,j} (A_{1j}A_{2j} - A_{2i}A_{2j})\, |ij\rangle$$

where $k = \sqrt{15}/4\sqrt{\pi}$,

$$d_{x^2-y^2} \xrightarrow{A} (A_{11}A_{12} - A_{21}A_{22})\, d_{xy}$$

$$+ (A_{11}A_{13} - A_{21}A_{23})\, d_{xz}$$

$$+ (A_{12}A_{13} - A_{22}A_{23})\, d_{yz}$$

$$+ \frac{1}{2}\,(A_{11}^2 - A_{21}^2 - A_{12}^2 + A_{22}^2)\, d_{z^2-y^2}$$

$$+ \frac{\sqrt{3}}{2}\,(A_{13}^2 - A_{23}^2)\, d_{z^2} \quad .$$

Finally,

$$n\, |2z^2 - x^2 - y^2\rangle \xrightarrow{A} n \sum_{i,j} (2A_{3i}A_{3j} - A_{1i}A_{1j} - A_{2i}A_{2j})\, |ij\rangle$$

where $n = \sqrt{15}/4\sqrt{\pi}$ yields

$$d_{z^2} \xrightarrow{A} \frac{1}{\sqrt{3}}(2A_{31}A_{32} - A_{11}A_{12} - A_{21}A_{22})\, d_{xy}$$

$$+ \frac{1}{\sqrt{3}}(2A_{31}A_{33} - A_{11}A_{13} - A_{21}A_{23})\, d_{xz}$$

$$+ \frac{1}{\sqrt{3}}(2A_{32}A_{33} - A_{12}A_{13} - A_{22}A_{23})\, d_{yz}$$

$$+ \frac{1}{2\sqrt{3}}(2A_{31}^2 - A_{11}^2 - A_{21}^2 - 2A_{32}^2 + A_{12}^2 + A_{22}^2)\, d_{x^2-y^2}$$

$$+ \frac{1}{2}(2A_{33}^2 - A_{13}^2 - A_{23}^2)\, d_{z^2} \quad .$$

The above equations are then used in ORBIT to calculate the 1×1, 3×3, and 5×5 matrices generated by each symmetry operation.

IV. POISON

POISON computes the positions of all atoms relevant to the problem at hand, and stores them in the array PNB. For example, if the program required interactions between atom 1 and up to next-nearest atoms of type 2 (N[1, 2] = 2), then POISON would compute the positions (relative to atom 1) of all atoms of type 2 nearest to 1 (assigning them an NX value of unity) and of all atoms of type 2 the next-nearest distance out (assigning them an NX value of 2).

Starting with the position in the unit cell of the "variable" atom (in the above case, atom 2), POISON generates a set of lattice sites using the lattice translations. The sites so generated outline a parallelepiped with $(2 \times NMAX) + 3$ sites on a side, where NMAX is the largest NX value to be considered. These positions are stored in the array PSN. This is a sufficiently large number such that, among these sites, we may always find the relevant sites to be stored in PNB.

POISON searches among the sites for the closest set. It stores the number of atoms in this closest set (in the array NNNX), the radial distance (in the array DNX), and their positions (in the array PNB). These sites are given an NX value of unity. Then, it sets the associated dummy positions PSN to 10^6, thus eliminating this set of lattice sites from consideration when POISON searches for the set of next-closest sites. Hence, the same search routine can be used again; that is, POISON again looks for the set of closest lattice sites. This set is then given an NX value of 2. This process continues until POISON has computed the positions for NMAX levels of atoms.

V. TFSX

LCAO picked a pair of atoms (NL, NM) and a corresponding pair of orbitals (LL, LLM, ML, MLM) which specify an entry in the secular determinant. Within a given entry, we must consider, in general, many transfer integrals corresponding to a ring of n. n., n. n. n., etc. Before parameterizing these transfer integrals, TFSX first computes the variables NO and NOX described below.

Recall that an arbitrary atom in the lattice is completely specified by a set of numbers such as NL, NM, NX, and NLEV. This atom lies within some ring of atoms, all of whom are the same distance from atom NL. Now, all these atoms may not have the same NM value; further, they may not even have the same NX value. For example, suppose there are two types of atoms in our ring: types 3 and 5. This ring may contain the closest type 3; hence, its NX value is unity. However, we may have intervening (closer) atoms of type 5, and its NX value may be 2. Similarly, there may be a different number of types 3 and 5 atoms in the ring. The subscripted variables NOX and NO give, respectively, for each type of atom in the ring, the level and the number of atoms. If there are no atoms of a given type, then NO = 0 for this type. These numbers (NO, NOX) are used later in ZERO (Sec. VI) in calculating to what position in the ring a given atom is transformed under some symmetry operation (since it is a point transformation, the distance from NL remains the same).

Before continuing with the flow of the program it would be well to make a digression. There are two separate outputs (all output statements other than error messages will be found in TFSX and NONSYM). The first information that appears on output 1 is the relationships between the various transfer integrals and their H.C. (this was discussed in Sec. I). The same notation is used in the output as is used in this report. Then, the secular determinant is printed out. A given entry is specified by the values of LL and ML, the chemical type atom, and the values of NL, NM, LLM, and MLM. Within the entry, there may be one or more parameters – each parameter corresponding to a transfer integral. A given parameter may appear more than once (in the same or different entry), indicating an equality between transfer integrals. If a parameter number is ever negative, then that parameter is equal to the negative of the parameter associated with the corresponding positive number, e.g., $P(-31) = -P(31)$. Further, associated with each parameter are the relative x-, y-, and z-coordinates of the NL and NM atoms (measured from atom NL to atom NM). One may then compute the structure factor associated with that parameter or transfer integral. For example, if we have

P(31) (1.00, 2.00, 0.0)

this corresponds to a structure factor $e^{i(k_x a + 2k_y a)}$ multiplying parameter 31. (Recall that all distances are given in units of some convenient length, a.) A blank entry indicates that all the transfer integrals in that entry are 0. Which transfer integrals correspond to which parameters, and which transfer integrals are 0 are indicated on output 2. Exactly how this is done will be amplified later in this section as we discuss ZERO, PARA, and HOPLES.

It should also be noted that the status of any given transfer integral (whether it has been parameterized or not, whether it is 0 or not) is indicated by the value of the subscripted variable ITF, which is initially set a -2×10^4 in LCAO. Hence, if some transfer integral has not been considered, its ITF value is -2×10^4; if found to be 0, its ITF is 0; if a nonzero parameter, its ITF value lies between $\pm 10^4$. If two transfer integrals are identical, they have the same ITF value; if they are the same except for a minus sign, the ITF values are the same except for a minus sign. Further usage of ITF will be explained as the need arises.

Proceeding with our program, TFSX calls ZERO to see if the particular transfer integral under study is 0. If so, it prints out as explained above; if not, TFSX calls PARA. PARA determines whether this transfer integral is equal to a linear combination of previously considered (hence, parameterized) transfer integrals. If so, there are various possibilities. If there is

only one nonzero parameter,* and the coefficient multiplying that parameter is ±1, then the ITF values are equated to each other or to the negative of each other, and that parameter with the corresponding coordinate of the NM atom is printed on output 1. If the coefficient is not equal to ±1, then (a) the relation between the transfer integrals is printed on output 2, (b) the given transfer integral is assigned a new parameter number, and (c) that parameter with the corresponding coordinate of the NM atom is printed on output 1. If there is more than one nonzero parameter, then TFSX proceeds in a fashion similar to the case when we have a single coefficient not equal to ±1.

PARA tries to relate the given transfer integral to linear combinations of parameters by looking at individual equations. (Recall that each symmetry transformation generates a new equation.) However, it may be possible to find an expression relating the given transfer integral to a linear combination of parameters by solving the entire set of equations generated by the transformations. This possibility is investigated in HOPLES (Sec. VIII). Hence, if PARA is not able to relate the given transfer integral to a linear combination of parameters, TFSX calls HOPLES. There is an exception, however. The variable XNOP keeps track of the number of parameters encountered in the set of equations generated by all the transformations. If XNOP is 0, then the number of parameters is 0. Hence, these equations are unsolvable and there is nothing to be gained in calling HOPLES. The given transfer integral is set equal to a new parameter, and this parameter with its corresponding position is printed on output 1. TFSX indicates which transfer integral this parameter number corresponds to on output 2.

HOPLES may find a set of linear equations which can be solved for the given transfer integral in terms of some parameters. This set is then printed on output 2, the given transfer integral is set equal to a new parameter, and this parameter with its corresponding position is printed on output 1. If HOPLES does not find such a set of linear equations, then TFSX sets the transfer integral equal to a new parameter and prints this relation on output 2. Further, on output 1 the corresponding entry in the secular determinant is made.

VI. ZERO

ZERO tests whether a given transfer integral is 0 or not. If the integral is 0, then it returns YESZ = 1; otherwise, it sets YESZ = 0. Essentially, it generates a set of equations using the NS symmetry operations. As indicated above, these are all of the form: Given transfer integral equals linear combination of transfer integrals. Then it tests these equations to find a pair where the respective right-hand sides are the negative of each other. If such a pair exists, the transfer integral is 0.

Before the above process takes place, ZERO performs another function. Having specified a transfer integral, we then have specified a pair of atoms, one of which is considered to be at the origin. A given .SSO. will transform the atom not at the origin into some other (or the same) atom. These .SSO. can be grouped according to the position to which the atom transforms. This information is used to save a considerable amount of storage space. It is also used for an entirely distinct reason in HOPLES (see Sec. VIII for further details).

A word about notation: KK or KKW is an array indicating how many .SSO. transform to some particular atom, and MIS or MISW is an array indicating the corresponding .SSO.

---

*Transfer integrals which have been previously considered, or parameterized, will be referred to as "parameters"; otherwise, they will be referred to as "variables."

VII. PARA

PARA looks for an equation of the form: Given transfer integral equals linear combination of parameters. It uses KK and MIS from ZERO to group the equations; then, PARA generates the equations in each set. Each transfer integral on the RHS undergoes a series of tests. First, PARA checks whether it is a parameter or a variable (by looking at the value of ITF). If it is a parameter, ITF is checked again to see if it is 0 or not; if it is 0, it is dropped from the right-hand side. The same test indicates whether it is the negative of some other parameter; if it is, the sign of the coefficient is changed and the parameter number is made positive. In either case, the variable (NOTZER) which counts the number of parameters on the RHS of this equation is increased by unity, and the coefficient and parameter number are loaded into two arrays (COEF and ITF2).

On the other hand, if the transfer integral is a variable, it goes through a different process. First, PARA checks whether it is proportional to the transfer integral on the LHS by generating a separate set of equations with the original transfer integral on the LHS. It scans this set for an equation of the form: Original transfer integral equals coefficient times this variable. If such an equation exists, it moves the variable to the RHS of our original equation by adjusting the value of ONE – the coefficient of our original transfer integral. Otherwise, PARA goes on to generate yet another set of equations. This time the variable is on the LHS. Each equation is tested to see if the RHS is a linear combination of parameters. If a variable is ever encountered on the RHS, that equation is immediately dropped. If it succeeds in this, then the variable is converted into the linear combination of parameters by loading the appropriate numbers into COEF and ITF2. If no such equation exists, then a variable NVAR counting the number of variables is increased by 1. After each transfer integral on the RHS of the original equation has gone through the above process, NVAR is tested. If NVAR is 0, PARA checks the value of ONE. If ONE $\neq 0$, PARA proceeds; but, if ONE is 0, PARA goes on to a new equation.* Assuming ONE is not 0, PARA then checks the value of NOTZER. If NOTZER and NVAR are both 0, then we have nothing on the RHS. Our transfer integral is 0, and YESZ is set = 1. If NOTZER ≥ 1, then YESLC is set = 1. This indicates to TFSX that PARA has succeeded in finding an equation relating the given transfer integral to a linear combination of parameters. PARA then transfers back to TFSX.

The other possibility is that NVAR is nonzero. PARA then checks NOTZER. If NOTZER is 0, i.e., we have all variables on the LHS, PARA considers a new equation; if it is not ZERO, then we have a mixture of variables and parameters on the RHS. The variable XNOP (to be explained below) is then increased by unity, and PARA continues to a new equation.

Finally, suppose PARA goes through the entire collection of sets of equations without transferring back to TFSX. The value of XNOP indicates whether or not all the equations are of the form: Transfer integral equals linear combination of variables only. In this case, there is no reason to call HOPLES. XNOP communicates this to TFSX. However, if XNOP > 0, then TFSX calls HOPLES.

VIII. HOPLES

The first half of HOPLES is essentially a duplicate of PARA, the only distinction being that instead of discarding a particular equation in a given set, HOPLES retains it. The coefficient

* If ONE is 0 and NVAR is 0, at best we have a relation among parameters which is not useful at the mament.

on the LHS is stored in the array ONEK for each equation in the set. Similarly, the arrays ITFV, COEFV, ITFP, COEFP, and NOTZR contain, respectively, the variable numbers, coefficients of these variables, parameter numbers, coefficients of parameters, and the number of parameters for each equation. Note that the variables are numbered internally within a given set of equations (the first variable encountered in a set is given the number 1, etc.).

HOPLES proceeds to look at this set of equations and selects the linearly independent ones. HOPLES checks whether a given equation is linearly dependent with respect to any of the other equations. If it is not, then it is stored as a linearly independent equation. This checking process is accomplished by comparing the given equation with the other equations in the set in the following manner:

(a) Ensuring there is at least one variable with a nonzero coefficient in the given equation. (If this is not the case, then we have a relation among parameters only, and the equation is discarded.)

(b) Finding the coefficient of the variable in the equation being compared.

(c) Taking the ratio of this coefficient with the corresponding coefficient in the given equation. (If the variable is missing in the equation being compared, the ratio is 0.)

(d) Comparing this ratio with the other ratios generated from corresponding coefficients.

Now, if the ratios are all equal, the two equations are linearly dependent; otherwise, they are linearly independent.

One should note that there is a special case where the ratios are not actually compared. Suppose some original ratio has been found; it will be 0 or some finite number. Now, if a subsequent ratio would involve dividing by 0, it is, of course, never computed and the two equations are linearly independent.

The array LIE indicates whether a given equation is linearly independent (LIE = 1) or not (LIE = 0). When the above process has been completed, the LIE's are summed. This gives the number of linearly independent equations.

This set of equations can fall into one of three categories: (a) number of equations (NUMEQ1) is less than the number of variables (NUMV + 1) where we include the transfer integral on the LHS as a variable; (b) NUMEQ1 = (NUMV + 1); and (c) NUMEQ1 > (NUMV + 1). In turn, this gives us three cases into which the sets fall: (1) all sets are of type (a); (2) some (possibly none) sets are of type (a), and some of type (b); and (3) at least one set is of type (c). For case (1), HOPLES returns the variable NUMEQ with the value 2×10^4 to indicate this; for case (2), it returns the smallest NUMEQ1 as the variable NUMEQ. Further, HOPLES returns the set of equations (variables, coefficients of variables, parameters, coefficients of parameters) and NUMVAR, the number of variables.

If a set of type (c) is ever encountered, HOPLES immediately transfers control back to TFSX along with the same information indicated after case (2). TFSX then prints out these equations with the message that they should be checked for consistency.

IX. NONSYM

If a given "fixed" atom I is related to a previously considered "fixed" atom J by an .NSSO., then LCAO calls NONSYM. The equation NSYM(I) = J carries this relationship. Its argument I equals the number of the atom being considered; its value J equals the number of the atom to

which the former is related by the given . NSSO. If there is no such relation, then NSYM(I) = 0. One must not use the . NSSO. that relates I to J until J has had all its transfer integrals parameterized. This means that the value of J must be less than the value of I. Hence, it is necessary to exercise some care in numbering the inequivalent atoms.

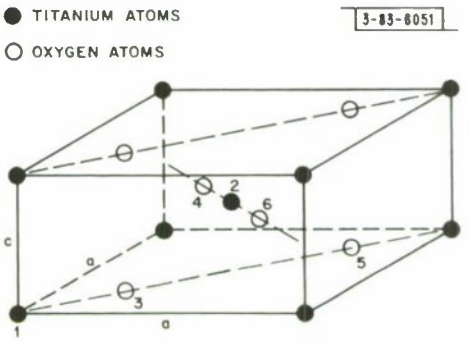

Fig. 4. TiO_2 rutile structure.

TiO_2 is a case in point; it has a rutile structure (see Fig. 4) with D_{4h} as its space group. If one chooses as the . NSSO. rotation by $-90°$ in the x-y plane, followed (or preceded) by the translation $(-\frac{a}{2} - \frac{a}{2} - \frac{c}{2})$, this then requires the numbering system indicated in Fig. 4, since $2 \rightarrow 1$, $6 \rightarrow 5$, $5 \rightarrow 4$, and $4 \rightarrow 3$ under the above . NSSO.

The first half of NONSYM consists of taking the relative coordinates between the given pair of atoms and rotating these coordinates, using the . NSSO. Then, NONSYM searches for an atom in that position relative to the corresponding previously considered "fixed" atom.

For example, in the above case when LCAO considers any transfer integral with atom 2 at the origin, it goes to NONSYM. There, the relative coordinates are rotated by $-90°$. Then, NONSYM searches for an atom with these coordinates relative to atom 1. Note that the atom not at the origin need not have the same inequivalent atom number in both cases (although, of course, it must be of the same chemical type).

Having located the corresponding atom not at the origin (which would be specified by a set of NM, NX, and NLEV values), NONSYM calculated the linear combination of transfer integrals generated by the . NSSO. It then tests each associated ITF value. If ITF is 0 or if the coefficient of that transfer integral is 0, then that parameter is not considered. If its ITF and coefficient are nonzero, then their values are stored in the arrays ITFNS and COEFNS, respectively. From here, the treatment is the same as in TFSX when the given transfer integral has been found to be a linear combination of parameters.

ACKNOWLEDGMENT

The authar acknawledges with pleasure the inspiratian and encouragement af Dr. J.M. Hanig.

DOCUMENT CONTROL DATA – R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Lincoln Laboratory, M.I.T.	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP None

3. REPORT TITLE

LCAO Secular Determinant Program

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Report

5. AUTHOR(S) *(Last name, first name, initial)*

Esterling, Donald

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
12 July 1967	20	1

8a. CONTRACT OR GRANT NO. AF 19(628)-5167 b. PROJECT NO. 649L c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report 435 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* ESD-TR-67-328

10. AVAILABILITY/LIMITATION NOTICES

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
None	Air Force Systems Command, USAF

13. ABSTRACT

This report describes a computer program designed to set up the secular determinant arising from an energy band calculation in the LCAO (Linear Combination of Atomic Orbitals) approximation. The program determines which transfer integrals vanish, and which are related; further, it computes the appropriate structure factor (which contains the momentum dependence) for each entry in the secular determinant. This program can handle arbitrary crystal symmetry, unit cells with many inequivalent atoms, interactions involving up to fourth-nearest neighbors, and a choice of s-, p-, and/or d-orbitals on the various inequivalent atoms. The transfer integrals are left as parameters to be determined from the eigenvalues corresponding to special symmetry points in the Brillouin zone.

14. KEY WORDS

computer programs
energy band theory – LCAO
secular determinant
tight binding approximation
LCAO approximation